International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-11 Issue-2, December 2021

GANs and VAEs As Methods of Synthetic Data Generation and Augmentation to Enhance Heart Disease Prediction

Rohit Sahoo, Vedang Naik, Saurabh Singh, Shaveta Malik

Abstract*: Heart disease instances are rising at an alarming rate, and it is critical and essential to predict any such ailments in advance. This is a challenging diagnostic that must be done accurately and swiftly. Lack of relevant data is often the impeding factor when it comes to various areas of research. Data augmentation is a strategy for improving the training of discriminative models that may be accomplished in a variety of ways. Deep generative models, which have recently advanced, now provide new approaches to enrich current data sets. Generative Models like Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs) are frequently used to generate high quality, realistic, synthetic data essential for machine learning algorithms as they play a critical role in various classification problems. In our case, we were provided with 304 rows of heart disease data to create a robust model for predicting the presence of an ailment in the patient. However, the identification of heart disease would not be efficient given the small amount of available training data. We used GAN, CGAN, and VAE to generate data to tackle this problem, thus augmenting the original data. This additional data will help in increasing the accuracy of the models created using the new dataset. We applied classification-based Machine Learning models such as Logistic Regression, Decision Trees, KNN, and Random Forest. We compared the accuracy of the said models, each of which was supplied with the original dataset and the augmented datasets that used the data generation techniques mentioned above. Our research suggests that using data generation techniques significantly boosts the accuracy of the machine learning techniques applied to them.

***Keywords*: Generative Adversarial Networks (GANs), Variational Autoencoders (VAEs), Conditional Generative Adversarial Networks (CGANs), Synthetic Data Generation**

I. INTRODUCTION

To develop a strong and accurate model, machine learning algorithms require a significant quantity of data to be analyzed. It would be impossible to generate such enormous quantities of data without synthetic data.

Manuscript received on November 15, 2021.
Revised Manuscript received on November 17, 2021.
Manuscript published on December 30, 2021.
* Correspondence Author
Rohit Sahoo*, Department of Computer Engineering, Terna Engineering College, Navi-Mumbai, India. Email: rohitsahoo741@gmail.com
Vedang Naik, Department of Computer Engineering, Terna Engineering College, Navi-Mumbai, India. Email: vedangnaik0812@gmail.com
Saurabh Singh, Department of Computer Engineering, Terna Engineering College, Navi-Mumbai, India. Email: srbhsingh39@gmail.com
Dr. Shaveta Malik, Associate Professor, Department of Computer Engineering, Terna Engineering College, Navi-Mumbai, India. Email: shavetamalik687@gmail.com

It would be impossible to generate such enormous quantities of data without synthetic data. Synthetic data is created to fulfill particular demands or situations that are not always available in the current data. This enables us to account for unanticipated outcomes and to have a rudimentary solution or cure in place in the event that the outcomes are unacceptable. Synthetic data is frequently created to reflect real data and to provide a baseline. In this paper, we will look at GANs and VAEs, two popular data generation techniques. Goodfellow et al. [1] developed generative adversarial networks (GANs), a strong family of generative models, in 2014. Generic modeling is a type of unsupervised machine learning job that automatically identifies and learns regularities or patterns in incoming data. The model can be used to produce fresh instances that might have been chosen from the original dataset. The core idea of GANs is based on a two-person zero-sum game, in which the profit of another player perfectly balances each player's loss. GANs are generally made up of a generator and a discriminator that both learn simultaneously [2]. The generator creates new data samples while attempting to capture the possible distribution of actual samples. A binary classifier is frequently used as the discriminator, separating actual samples from produced samples as precisely as feasible. The generator and discriminator can be built using neural networks. GANs' optimization process is a minimax game process, intending to reach Nash equilibrium when the generator is thought to have captured the distribution of actual samples. VAE is a generative model that identifies dimensional relationships and generates fresh samples similar to but not identical to those in the original dataset. To create new samples, VAE follows a two-step procedure. To begin, it selects those latent variable values that are most likely to provide data. The conditional distribution of the data, given the latent variable, is then used to create fresh samples. The mathematical foundation of VAE differs significantly from that of a standard autoencoder. Neural networks are often used to build the encoder and decoder [3]. The encoder is used to produce the latent variable's estimated posterior to understand the complex data distribution. The latent variable's values are then sent into the decoder, which produces new samples. VAE is trained by minimizing its loss function, composed of two halves, using gradient descent [4]. The reconstruction error between the original and produced sample is the first component. The Kullback-Leibler KL divergence between the latent variable's estimated posterior and actual prior is the other component.

The latent variable's value is sampled from the previous distribution following the training procedure. The sampled value is then sent into the decoder, which produces a fresh sample. The second section describes the data generation methods that we have used. The third section describes how we have used the newly generated data to classify whether a patient has heart disease. The fourth section goes over the results of our experiments. Finally, the fifth section concludes the paper.

II. DATA GENERATIVE METHODS

A. Generative Adversarial Networks (GANs)

GAN's central concept is derived from the Nash equilibrium in game theory. It presupposes two players in the game: a generator and a discriminator. The generator's goal is to learn the composition of accurate data, whereas the discriminator's goal is to correctly identify whether the input data is actual or synthetic. To succeed, the two players must constantly increase their abilities. The goal of this process is to arrive at a Nash equilibrium between the two parties. The computation procedure and structure of GAN are shown in (Fig. 1).

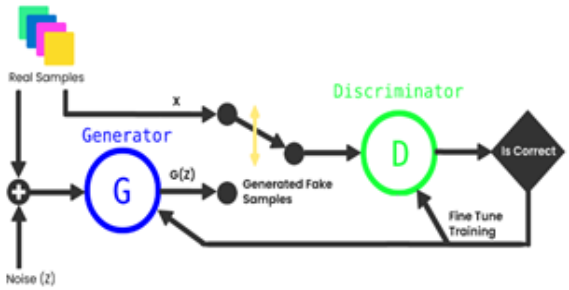

Fig. 1. Generative Adversarial Network

The discriminator and generator are defined by differentiable functions D and G, and their inputs are actual data x and random variables z, respectively. G(z) is a sample created by G that follows the distribution of actual data (p_{data}). If discriminator D receives input from real data x, D should categorize it as true and label it as 1. If the input comes from G(z), D should label it as 0 and categorize it as false. The goal of D is to correctly identify the data source, whereas the goal of G is to ensure that the effectiveness of produced data G(z) on D (i.e., D(G(z)) is consistent with the effectiveness of actual data x on D (i.e., D(x)). The generator's objective is to generate realistic data samples based on a random variable z ~ p_z (z), while the discriminator's job is to discriminate between actual data samples x~p_{data}(x) and produced samples p_g(x) ~ (x) [5]. The adversarial optimization method steadily enhances D and G's performance. It is assumed that generator G has reproduced the pattern of actual data when D's identification skill has increased significantly, but it still cannot differentiate the data source properly. D wants to enhance the possibility of actual samples while decreasing the chance of creating samples. G, on the other side, desires that D maximize the possibility of created samples. When D can no longer differentiate between genuine and produced samples, a Nash equilibrium state is achievable [6].

Initially, we'll go through how to optimize discriminator D given generator G. Training the discriminator is analogous to training Sigmoid function-based classifiers in that it requires reducing cross-entropy. The following is the formula for the loss function:

$$Obj^{D}(\theta_D, \theta_G) = -\frac{1}{2}E_{x\sim p_{data}(x)}[logD(x)] - \frac{1}{2}E_{z\sim p_z(z)}\left[log\left(1 - D\left(g(z)\right)\right)\right] \quad (1)$$

Where x is taken from the actual data distribution pdata(x), z is taken from a prior distribution p_z(z), such as a uniform or Gaussian distribution, and E(·) is the expected value. Thus, the training data is made up of two parts: one from the real data distribution p_{data}(x) and the other from the produced data distribution p_g(x) (x). This differs somewhat from traditional binary classification algorithms. To find the best solution, we must minimize (1) given the generator.

$$D_G^*(x) = \frac{p_{data}(x)}{p_{data}(x) + p_g(x)} \quad (2)$$

This is discriminator D's best solution. GAN's discriminator calculates the ratio of two probability densities based on the above formula. The probability of x taken from real data rather than manufactured data is denoted by D(x). The discriminator attempts to make D(x) approach 1 if the input data is from genuine data x. If the input data is produced data G(z), the discriminator attempts to get D(G(z)) as close to 0 as possible, while the generator G tries to get it as close to 1. Because G and D are playing a zero-sum game, G's loss function is $Obj^{G}(\theta_G) = -Obj^{D}(\theta_D, \theta_G)$. As a result, GAN optimization may be expressed as a minimax problem [7]:

$$min_G max_D\{f(D,G) = \frac{1}{2}E_{x\sim p_{data}(x)}[logD(x)] + \frac{1}{2}E_{z\sim p_z(z)}[log(1 - D(G(z)))]\} \quad (3)$$

In order to train GANs, we must train both generator and discriminator networks. Across the training process, both actual data collected from a dataset and false data produced continually by the generator are used. The discriminator is taught similarly to a neural network classifier. The discriminator is taught to allocate data to the "actual" class in this scenario. The false data is made by choosing a random vector z from a previous distribution over the model's latent variables. After that, the generator is used to generate a sample x = G(z). The function G is essentially a neural network-based function that converts the random, structureless z vector into structured values similar to the training data. The discriminator then assigns a classification to this fictitious data. The backpropagation algorithm allows the generator to be trained by using the derivatives of the discriminator's output about the discriminator's input. The generator is programmed to deceive the discriminator, causing it to allocate its input to the "actual" class. The discriminator's training procedure is similar to a binary classifier, except that the input for the "fake" class comes from a continually changing data spread rather than a fixed data spread as the generator learns.

The generator's learning process is unusual in that it is not given explicit output objectives but instead is rewarded for creating outputs that deceive its (forever evolving) opponent.

B. Conditional Generative Adversarial Networks (CGANs)

Generative adversarial networks can be expanded to a conditional model if both the generator and the discriminator are conditioned on some auxiliary information y, such as feature labels or input from other modalities. The previous input noise $p_z(z)$ and y are merged in a joint hidden representation in the generator, and the adversarial training leads to substantial flexibility in the composition of this hidden representation [8].

$$min_G max_D V(D,G) = E_{x\sim p_{data}(x)}[logD(x|y)] + E_{z\sim p_z(z)}[log(1 - D(G(z|y)))] \quad (4)$$

The objective is to resample effectively during the training phase to sample all discrete attribute categories equally (but not necessarily uniformly) and recover the (non-resampled) true data distribution during the test. If k^* be the value from i^*th discrete column D_{i*} that must be coordinated by the produced samples $\hat{r}$, the generator may be thought of as the conditional distribution of rows given that particular value in that particular column, i.e., $\hat{r} = P_G(row|D_{i*} = k^*)$. Hence, this structure is called Conditional GAN. The following concerns must be addressed when integrating a conditional generator into a GAN's design:

$$P_G(row|D_{i*} = k^*) = P(row|D_{i*} = k^*) \quad (5)$$

so, we can regenerate the original distribution as:

$$P(row) = \sum_{k\in D_{i*}} P_G(row|D_{i*} = k^*)P(D_{i*} = k^*) \quad (6)$$

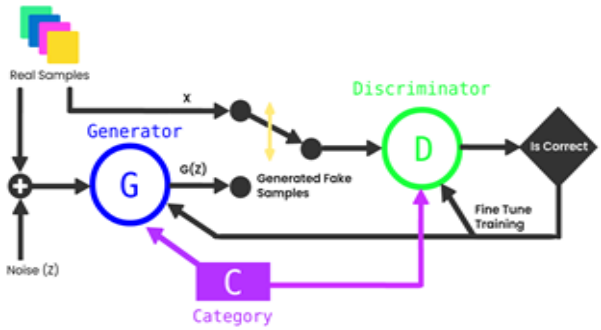

Fig. 2. Conditional GAN

C. Variational Autoencoder (VAE)

A variational autoencoder (VAE) has an architecture similar to an autoencoder that combines a directed probability-based graphical model within a neural network. An encoder and a decoder are the two fundamental components of a VAE model. Given observed data x, the encoder initially learns a conditional distribution of a latent variable z. The decoder then reconstructs data x from the latent code generating distribution [9].

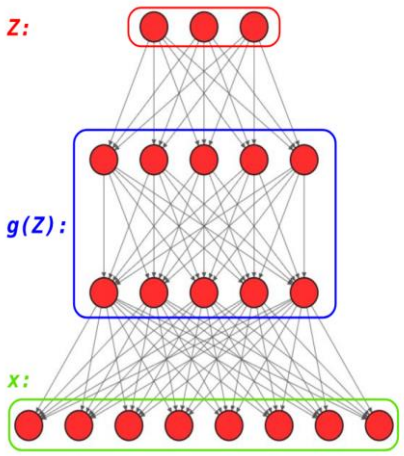

Fig. 3. VAE Directed Graphical Model

The uppermost layer of model z in (Fig. 3) is the latent variable where the producing process in the VAE begins. In (Fig. 3), g(z) represents the complex data generation process, structured as a neural network. The marginal likelihood is problematic; the objective function is the variational lower bound of the data having marginal probability. This probability is the total of the marginal likelihoods of all data provided:

$$logp_\theta(x^{(1)},\ldots,x^{(N)}) = \sum_{i=1}^{N} logp_\theta(x^{(i)}) \quad (7)$$

Where the data points can be represented as: [10]

$$logp_\theta(x^{(i)}) = D_{KL}(q_\phi(z|x)||p_\theta(z)) + \angle(\theta,\phi;x^{(i)}) \quad (8)$$

The approximate posterior is qφ(z|x), while the previous distribution of the latent variable z is p(z). The above equation can be restated as follows since the Kullback-Leibler divergence component is always greater than zero [11]:

$$logp_\theta(x^{(i)}) \geq \angle(\theta,\phi;x^{(i)}) = E_{q\phi(z|x^{(i)})}[-logq_\theta(z|x) + logp_\theta(x|z)] = -D_{KL}(q_\phi(z|x^{(i)})||p_\theta(z)) + E_{q\phi(z|x^{(i)})}[logp_\theta(x|z)] \quad (9)$$

The probability of the input x given the z is pθ(x|z), and the Kullback-Leibler divergence between the previous of the z and posterior distribution is the first part given in the above equation. The regularization component compels the Approximate Posterior (AP) to be comparable to the previous distribution. The second part of the equation (9) may be explained using the AP qφ(z|x) and the probability pθ(x|z) to reassemble x. The VAE uses a neural network to represent the features of the AP qφ(z|x). The AP qφ(z|x) is the encoder in the concept of autoencoder, and the DPGM pθ(x|z) is the decoder,

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

as illustrated in (Fig. 4). That is, the encoder's f(x, φ) produces the parameter of the estimated posterior qφ(z|x), and sampling from q(z; f(x, φ)) is necessary to retrieve the real value of the latent variable z. As a result, VAE's encoders and decoders are known as probabilistic encoders and decoders.

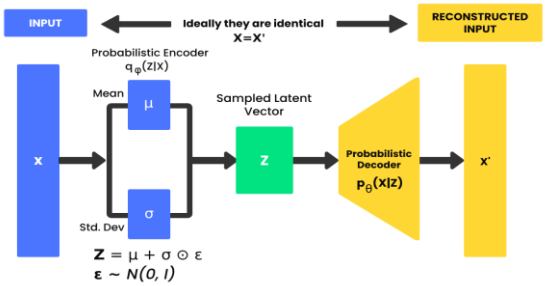

Fig. 4. Variational Autoencoder

III. EXPERIMENTS

The suggested system method aims to increase the accuracy of forecasting cardiac disease by utilizing multiple techniques. The architecture of the said system is described in Fig. 4. It consists of the following stages: data collection, data generation, data analysis, feature engineering, model training.

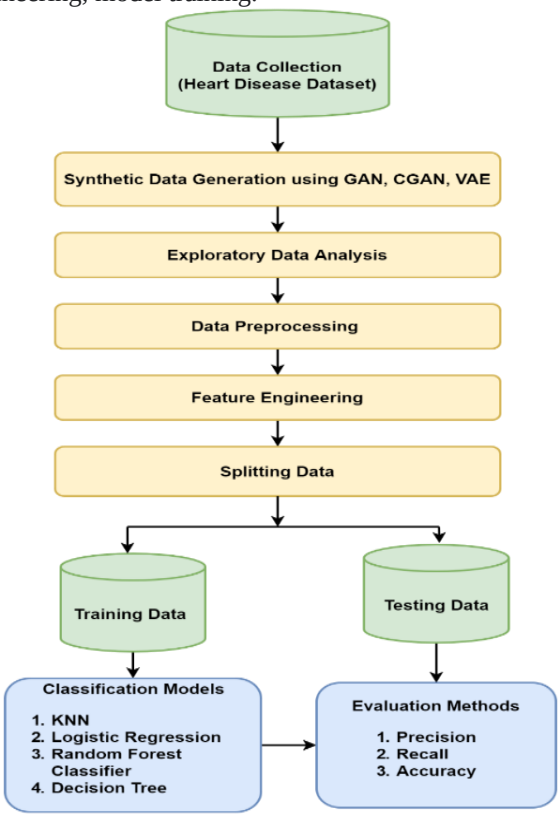

Fig. 4. Flowchart of the experiment

The suggested approach's phases are outlined in full below.

A. Dataset

Individuals were chosen from an organized dataset based on their history of cardiac issues as well as other medical disorders. Cardiovascular illnesses are the leading cause of mortality among middle-aged individuals, according to the World Health Organization (WHO). We use a data set that includes the medical histories of 304 individuals ranging in age from 18 to 84 years old.

This dataset provides us with critical information, such as the patient's age, resting blood pressure, fasting sugar level, and other medical characteristics, which aids us in determining whether or not the patient has been diagnosed with heart disease.

This dataset comprises 13 medical characteristics for 304 individuals that help us determine whether or not the patient is a danger of developing heart disease and identify patients who are at risk and those who are not. This Heart Disease dataset comes from the University of California at Irvine's repository.

The structure that leads to identifying people at risk for heart disease is taken from this dataset. There are two components to these records: training and testing. Each row corresponds to a single record in this dataset, which has 304 rows and 14 columns. 'Table 1' has a list of all characteristics.

Table- I: Various Attributes Used Are Listed

Observation	Description	Values
Age	Age in years	Continuous
Sex	Sex of Subject	Male/Female
CP	Chest Pain	Four Types
Trestbps	Resting Blood Pressure	Continuous
Chol	Serum Cholesterol	Continuous
FBS	Fasting Blood Sugar	<, or> 120 mg/dl
Restecg	Resting Electrocardiograph	Five Values
Thalach	Maximum Heart Rate Achieved	Continuous
Exang	Exercise-Induced Angina	Yes/No
Oldpeak	ST Depression when Workout compared to the Amount of Rest Taken	Continuous
Slope	The slope of the Peak Exercise ST segment	Up/Flat/Down
Ca	Gives the number of Major Vessels Colored by Fluoroscopy	0-3
Thal	Defect Type	Reversible/Fixed/Normal
Num (Disorder)	Heart Disease	Not Present/Present in the Four Major types.

B. Data Generation

Adversarial techniques generate augmented instances by introducing adversarial modifications to the original data, which significantly impacts the model's forecasts and reliability without affecting human judgment [12]. In our experiment, we have used GAN, CGAN, and VAE to generate synthetic data and augmented it to the actual dataset.

GAN and CGAN

The generator and the discriminator are defined as feed-forward networks with two hidden and fully connected layers, with the number of layers defined as a parameter for the experiments. The output layer for the discriminator is simply a fully connected layer with just one unit. The generator's output layer, on the other hand, is affected by four scenarios involving two variables:

Retrieval Number: 100.1/ijeat.B32631211221
DOI: 10.35940/ijeat.B3263.1211221
Journal Website: www.ijeat.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

whether the model includes a conditional vector and whether the data to produce is categorical or continuous. As a result, two generator models were created.

The output is a fully connected layer with a ReLU activation and N units if the model is designed to create discrete data from the original patient data. The structure is the same when the model generates continuous data, but the activation function must be SoftMax. There is an extra layer for models with conditional data that concatenates the conditional vector with the prediction and outputs this concatenation. Batch-normalization and the ReLU activation function are used in the generator. The synthetic data representation is created using a combination of activation functions after two hidden layers.

The synthetic row representation is created using a combination of activation functions after two hidden layers. On each hidden layer, we apply the leaky rectified linear activation function and dropout in critic. We produced 300 unique patient data points from the original dataset of 304 data points in this study.

Variational Autoencoder (VAE)

VAE can synthesize new data, Z = g(z), by sampling in the low-dimensional space z. Two layers of feed-forward neural networks make up the encoder. The encoder's first and second layers each have a dense layer and a rectified linear activation function.

The data is flattened and sent to the decoder for upsampling in the final dense layer. The up-sampling layer, which comprises filters, is the first layer of the decoder. A dense layer neural network with rectified linear activation is the second layer. Finally, the last layer has a Rectified linear activation, a linear function that outputs the input directly if it is positive and zeros otherwise. In this experiment, we used VAE to create 300 unique patients using the original dataset, which included 304 data points.

C. Data Analysis

We analyzed the data in our experiment to better understand the distribution of original data and the differences between the distributions created by GAN, CGAN, and VAE. The exhaustive data analysis was used to identify the overall distribution of the data after augmenting the synthetic data with the original data.

The selection of the optimal features is critical since irrelevant features frequently influence the classifier's effectiveness. During this stage, linear discriminant analysis (LDA) and principal component analysis (PCA) pick important characteristics from the data. Fig. 5, Fig. 6, Fig. 7, and Fig. 8 show the data distribution of original data and the synthetic data generated using GAN, CGAN, and VAE, respectively, for the types of chest pain within patients having heart disease.

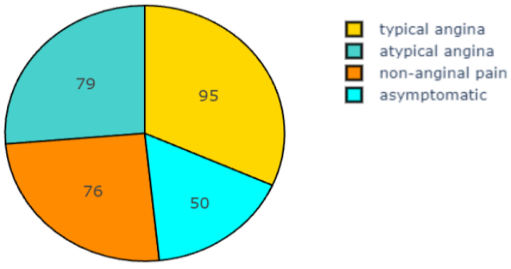

Fig. 5. Types of chest pain within patients having heart disease in the original dataset

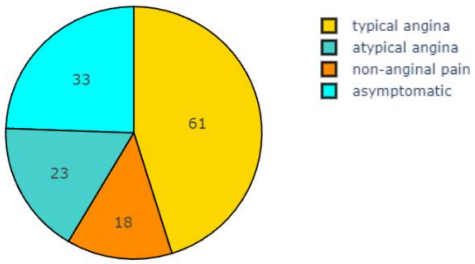

Fig. 6. Types of chest pain within patients having heart disease generated by GAN

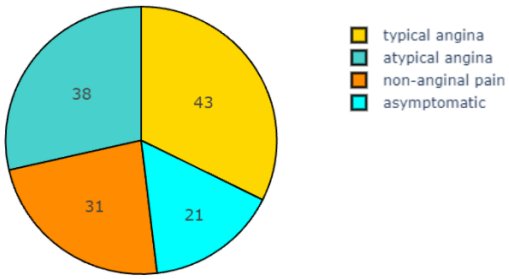

Fig. 7. Types of chest pain within patients having heart disease in generated by CGAN

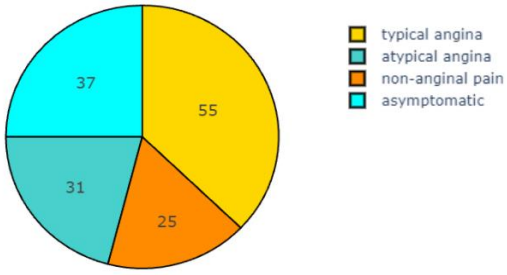

Fig. 8. Types of chest pain within patients having heart disease generated by VAE

D. Data Preprocessing and Feature Engineering

The feature engineering method entails encoding all of the dataset's properties such that any machine learning or statistical model can understand and interpret them effectively. There are several phases to this procedure. The first step is to identify any characteristics in the dataset with skewed values and execute any appropriate modifications (e.g., logarithmic or cube root).

Next, the continuous attributes are normalized to have a mean of zero and a standard deviation of one, understanding that having standardized values of a dataset is a prerequisite for many machine learning models –since they may behave adversely if the data does not seem customarily distributed data. When categorical values are involved, they are also represented as one-hot vectors. Finally, the features are adjusted to be between [0, 1]. It should be noted that missing data are removed from the dataset. Then, the heart disease dataset is partitioned into a 75% training set and a 25% testing set in this stage.

E. Model Training

After the data has been pre-processed, a classifier is used to categorize the pre-processed data, which is then added to the synthetic data created by the GAN, CGAN, and VAE.

The data is attached to the original dataset after the synthetic data created by the generative models are added, bringing the total number of data points to 604 (604 individual patients, 13 features, and 1 Target variable).

Before training any machine learning model, a standard scalar is used to normalize the features in each of the three datasets generated by each of the generative models, i.e., normalize each column of the three datasets separately so that each feature has a mean of zero and a standard deviation of one.

Then, the dataset is randomly split into two parts: 80 percentage for training and 20 percentage for testing. KNN, Logistic Regression, Random Forest Classifier, and Decision Trees were the classifiers utilized in our experiment.

IV. OBSERVATIONS AND RESULTS

In this experiment, we used assessment measures such as precision, recall, and accuracy to evaluate all machine learning models trained on actual and synthetic data generated by GAN, CGAN, and VAE.

A. True Positive (TP):

True Positives are situations in which the patients had the cardiac disease, and our model also predicted that they would have it.

B. True Negative (TN):

True Negatives are situations in which the patients did not have cardiac disease, and our model also predicted that they would not have it.

C. False Positive (FP):

When the patient does not have cardiac disease, but our model predicts that they will. This type of error is known as a Type I Error, and the values are referred to as False Positives.

D. False Negative (FN):

When the patient has cardiac disease, but our model predicts that he or she does not. This is known as a Type II Error, and the values are referred to as False Negatives.

E. Evaluation Methods:

The suggested model is evaluated using many criteria, including precision, recall, and accuracy.

F. Precision:

The proportion of Positive Instances to all Positives is known as precision. So, our experiment would be the proportion of patients with heart disease who we correctly identify out of all those who have it.

$$precision = \frac{TP}{TP + FP} \quad (10)$$

Precision also provides us with a count of the essential observations. Thus, for example, we must not begin treating a patient with a cardiac condition but were projected to have one by our algorithm.

G. Recall:

The recall is a measure of how well our model identifies True Positives. As a result, recall informs us how many patients we properly recognized as having heart disease out of all those who have it. Mathematically:

$$Recall = \frac{TP}{TP + FN} \quad (11)$$

The recall is a metric that indicates how well our model can detect relevant data. It is also known as True Positive Rate or Sensitivity.

H. Accuracy:

The ratio of the overall number of correct forecasts to the total number of predictions is known as accuracy.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \quad (12)$$

The precision, recall, and accuracy for Logistic Regression, Decision Trees, Random Forest, and KNN are shown in the tables below.

Table- II: Results of Logistic Regression

Name	Real Data	GAN	VAE	CGAN
Precision	0.83	0.86	0.90	0.93
Recall	0.77	0.81	0.84	0.87
Accuracy	0.80	0.84	0.87	0.90

Table- III: Results of Decision Tree

Name	Real Data	GAN	VAE	CGAN
Precision	0.76	0.78	0.85	0.86
Recall	0.78	0.81	0.79	0.81
Accuracy	0.77	0.79	0.81	0.84

Table- IV: Results of Random Forest

Name	Real Data	GAN	VAE	CGAN
Precision	0.83	0.91	0.94	0.95
Recall	0.74	0.74	0.81	0.85
Accuracy	0.80	0.83	0.86	0.88

Table- IV: Results of KNN

Name	Real Data	GAN	VAE	CGAN
Precision	0.67	0.67	0.70	0.71
Recall	0.62	0.70	0.64	0.68
Accuracy	0.71	0.72	0.73	0.76

V. CONCLUSION AND FUTURE SCOPE

Thus, in this paper, we have looked at three methods of synthetic data generation, namely GANs, CGANs, and VAE. We used these methods to generate synthetic data from a heart disease dataset and utilized the newly created dataset to predict the presence of the malady in a patient using the ML algorithms of Logistic Regression, Decision Trees, KNN, and Random Forest. In the future scope, we can implement robust machine learning solutions using ensemble methods.

Retrieval Number: 100.1/ijeat.B32631211221
DOI: 10.35940/ijeat.B3263.1211221
Journal Website: www.ijeat.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

REFERENCES

1. I.J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative Adversarial Networks, 2014.
2. X. Gong, S. Chang, Y. Jiang, and Z. Wang, AutoGAN: Neural Architecture Search for Generative Adversarial Networks. 2019.
3. S. Ding and R. Gutierrez-Osuna, "Group Latent Embedding for Vector Quantized Variational Autoencoder in Non-Parallel Voice Conversion," in Proc. Interspeech 2019, 2019, pp. 724–728. doi: 10.21437/Interspeech.2019-1198.
4. H. Dai, Y. Tian, B. Dai, S. Skiena, L. Song, Syntax-Directed Variational Autoencoder for Structured Data, 2018.
5. Triastcyn, B. Faltings, Generating Differentially Private Datasets Using GANs, 2018.
6. A.-A.-Z. Imran, D. Terzopoulos, Multi-Adversarial Variational Autoencoder Networks, 2019.
7. P. Gmarova, K.Y. Levy, A. Lucchi, N. Perraudin, I. Goodfellow, T. Hofmann, A. Krause, A domain agnostic measure for monitoring and evaluating GANs, 2020.
8. L. Chen, S.-Y. Lin, Y. Xie, H. Tang, Y. Xue, Y.-Y. Lin, X. Xie, W. Fan, TAGAN: Tonality Aligned Generative Adversarial Networks for Realistic Hand Pose Synthesis, in: BMVC, 2019.
9. H. Shao, J. Wang, H. Lin, X. Zhang, A. Zhang, H. Ji, T. Abdelzaher, Controllable and Diverse Text Generation in E-Commerce, in: Proceedings of the Web Conference 2021, Association for Computing Machinery, Ljubljana, Slovenia, 2021: pp. 2392–2401.
10. J. Xu and G. Durrett, Spherical Latent Spaces for Stable Variational Autoencoders. 2018.
11. J. Aneja, A. Schwing, J. Kautz, A. Vahdat, NCP-VAE: Variational Autoencoders with Noise Contrastive Priors, 2020.
12. J. Chen, D. Tam, C. Raffel, M. Bansal, and D. Yang, An Empirical Survey of Data Augmentation for Limited Data Learning in NLP. 2021.

AUTHORS PROFILE

Rohit Sahoo is an engineer with a bachelor's degree in Computer Engineering from the Terna Engineering College, University of Mumbai, Navi-Mumbai, India. His research interests are in Machine Learning, Deep Learning, Data Science and Big Data.

Vedang Naik is an engineer with a bachelor's degree in Computer Engineering from the Terna Engineering College, University of Mumbai, Navi-Mumbai, India. His research interests are in Deep Learning, Natural Language Processing, Neural Networks and Machine Learning.

Saurabh Singh is an engineer with a bachelor's degree in Computer Engineering from the Terna Engineering College, University of Mumbai, Navi-Mumbai, India. His research interests are in Data Mining, Machine Learning, Recommender Systems, and Data Science.

Dr. Shaveta Malik is working as Associate Professor at Computer Engineering Department at Terna Engineering College, Navi-Mumbai, India. She has presented several papers in international conferences. Her research interests are in Artificial Intelligence, Machine Learning and Image Processing.